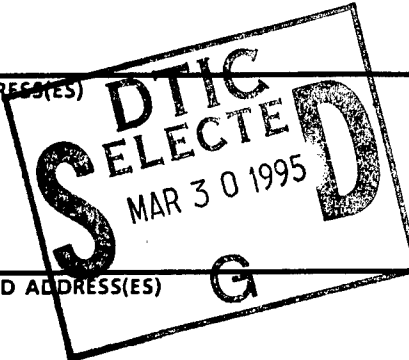# REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 25, 1995	Final 1 Aug 90 - 30 Sep 94

4. TITLE AND SUBTITLE

Research and Development of a High Performance
GaAs Microcomputer System

5. FUNDING NUMBERS

DAAL03-90-C-0028

6. AUTHOR(S)

Richard B. Brown

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Michigan
2236 EECS, 1301 Beal Ave.
Ann Arbor, Michigan   48109-2122

DTIC
SELECTED
MAR 3 0 1995
G

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U. S. Army Research Office
P. O. Box 12211
Research Triangle Park, NC   27709-2211

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ARO 28325.50-EL

11. SUPPLEMENTARY NOTES

The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The object of this research was to achieve an order of magnitude increase in RISC microprocessor computing power over the duration of the project. This was to result in a processor capable of executing MIPS instructions at a sustained rate of 150 million instructions per second, and a peak rate of 250 million instructions per second. We proposed implementing the machine in gallium arsenide direct-coupled FET logic (DCFL) from Vitesse Semiconductor, which showed promise as a high-performance technology. To achieve this goal, the GaAs process would have to be increased substantially over their levels at the beginning of the project. Circuits would have to be designed to take advantage of the inherent speed in GaAs devices while overcoming the limitations of DCFL. Microarchitectures would have to be developed to be appropriate for a processor with this high clock rate. We also proposed developing a comprehensive suite of CAD tools to support the design of such processors. This would include architectural, logic, and circuit simulators; it would build on the technology-independent physical design software of Cascade Design Automation to realize a GaAs circuit compiler.

14. SUBJECT TERMS

Gallium Arsenide, Microprocessor, MIPS, CAD

15. NUMBER OF PAGES
30

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Research and Development of a High Performance GaAs Microcomputer System

Final Report

Author: Richard B. Brown

January 25, 1995

U. S. Army Research Office

Award Number: DAAL03-90-C-0028 / ARPA Order Number: 7509

The University of Michigan

Foreword

This project was undertaken to help evaluate and demonstrate the potential of gallium arsenide digital VLSI circuits. The success of this effort has hinged on development of appropriate circuit designs, chip architectures, and design automation tools for GaAs.

In the circuits area, the project has contributed significant new designs for GaAs SRAMs, I/O pads, adders, mux-latch-buffers, and high-speed chip-to-chip interfaces, as well as a new logic family which improves the power-delay product for many commonly used digital circuits. Since the strengths and weaknesses of GaAs are different than those of CMOS, novel circuit topologies were used to maximize performance. A general approach for optimizing circuits for process tolerance was developed, which will also have applications in CMOS as channel lengths are scaled further; this optimizer has been incorporated into a GaAs SRAM compiler.

Microprocessor architectures for GaAs implementation have been studied, and three GaAs microprocessors (in addition to SRAMs and other test chips) have been designed. Two of the microprocessor designs have been fabricated and tested; the final design, which awaits fabrication, incorporates many advanced architectural features: two full integer pipelines, three independent floating-point functional units, out-of-order completion, branch prediction, prefetching for instructions and data and a non-blocking memory system.

The project has produced architectural simulation tools which are used to evaluate design tradeoffs; performance monitoring tools which illuminate the interdependencies of processor architecture and operating systems; delay macromodels for GaAs logic gates and on-chip and MCM interconnect; and physical design tools for GaAs circuits, including a design environment which compiles DCFL circuits from hardware-description-language input. This tool set has been commercialized and is in use at a number of US companies and universities.

2

Contents

Accesion For		
NTIS CRA&I	☒	
DTIC TAB	☐	
Unannounced	☐	
Justification		
By		
Distribution /		
Availability Codes		
Dist	Avail and / or Special	
A-1		

1 Research Objectives

The object of this research was to achieve an order of magnitude increase in RISC microprocessor computing power over the duration of the project. This was to result in a processor capable of executing MIPS instructions at a sustained rate of 150 million instructions per second, and a peak rate of 250 million instructions per second.

We proposed implementing the machine in gallium arsenide direct-coupled FET logic (DCFL) from Vitesse Semiconductor, which showed promise as a high-performance technology. To achieve this goal, the GaAs process would have to be controlled much better, and integration levels would have to be increased substantially over their levels at the beginning of the project. Circuits would have to be designed to take advantage of the inherent speed in GaAs devices while overcoming the limitations of DCFL. Microarchitectures would have to be developed to be appropriate for a processor with this high clock rate.

We also proposed developing a comprehensive suite of CAD tools to support the design of such processors. This would include architectural, logic, and circuit simulators; it would build on the technology-independent physical design software of Cascade Design Automation to realize a GaAs circuit compiler.

2 Research Results

Brief summaries of the most important research results are included in this section. A much more complete view of the results is available in the 50 papers and articles listed in the Publications section.

To summarize, we evaluated GaAs DCFL technology in real microprocessor designs. Close relationships were developed between the University of Michigan and Vitesse, Motorola, and Cray Computer; circuits were designed in all of their processes. We have also interacted with Tera Computer

Because DCFL has different strengths and constraints than CMOS, we developed circuits and architectures which were appropriate for the technology. A significant increase in performance and performance/power resulted from these innovations. During the course of the project, Vitesse improved the integration levels of DCFL circuits by an order of magnitude, solving a number of yield problems along the way.

Among the most important tangible results of the project are:

- A CAD environment for designing high performance GaAs VLSI circuits. This includes a circuit compiler which converts hardware description language input to layout and generates an accurate simulation model of the circuit. These tools have been commercialized.

- A methodology for macromodeling delays in gates and interconnect which is rigorous and computationally efficient.

- A methodology and CAD tools (Trace-Driven and Trap-Driven simulators) for evaluating the interdependencies of operating systems and hardware.

- An automated approach for optimizing circuit design in view of process, temperature, and voltage variations. Extensive use was made of this capability in the design

of our circuits to achieve robustness over process variations.

- Novel circuit designs for GaAs SRAMs, I/O pads, adders, mux-latch-buffers, and high-speed chip-to-chip interfaces, and a new logic family which improves the speed-power product for many commonly used digital circuits.

- A GaAs circuit compiler which has now been commercialized. Circuits are realized as physical datapaths and blocks of standard-cell gates. Placement, routing, and buffer sizing are performance-driven.

- A microprocessor architecture optimized for high clock rates, and appropriate for GaAs DCFL. It includes two full integer pipelines, three independent floating-point functional units, out-of-order completion, branch prediction, prefetching for instructions and data, and a non-blocking memory system.

- Eight Ph.D. and eleven M.S. graduates who are working in key faculty (Stanford and Carnegie Mellon) and industrial (Intel, Motorola, Hewlett-Packard, etc.) positions.

In addition, the work of this project has influenced technology directions taken by several major U.S. corporations.

Some of the important results of this project are summarized in the following sections.

2.1   Macromodeling of Gate and Interconnect Delays

A timing macromodel for GaAs DCFL logic gates has been derived. It calculates the delay of a gate as a function of such parameters as transistor sizes, capacitive loading, fanout, and input transition time. For NOR gates, the simultaneous switching of two inputs is also considered. Calculations based on the derived macromodel show excellent agreement with circuit simulation.

The development of a general strategy for delay computation based on dimensional analysis was the most important contribution of our macromodeling work. We have demonstrated the power of dimensional analysis as an indispensable tool in delay macromodeling, and have shown how to apply it in a systematic way for general behavioral macromodeling applications. When dimensional analysis is used in conjunction with circuit simulation and the Monte Carlo method, the resulting macromodels provide excellent accuracy without compromising efficiency.

The second general contribution is context delay modeling and the inclusion of transition time effects in timing analysis. The long path delays obtained for a front-end two-context gate delay model were within a few percent of circuit simulation, indicating the accuracy of both the context modeling approach and the gate macromodels.

We developed the first GaAs DCFL macromodel that takes into account the effects of input rise/fall times, nonlinear Schottky diode current, and input proximity. The accuracy and simplicity of the GaAs DCFL macromodels stem directly from the application of dimensional analysis. We have also derived CMOS macromodels by first calculating a general voltage waveform at the output of a CMOS inverter, and then applying dimensional analysis, which allowed us to extend the inverter macromodel to general CMOS static gates. The CMOS inverter macromodel is one-dimensional for purely capacitive loads. For RC loads, which model the loading of RC interconnect on the driving inverter, the macromodel is two-dimensional.

A general timing macromodel for RC interconnections was also developed. It takes into account the threshold levels at which the delay is measured, and the input rise/fall times. The macromodel was obtained after approximating the transfer function by a one-zero, two-pole function. The accuracy of the interconnect model was demonstrated, indicating the validity of the assumptions that we made.

Off-chip lossy and lossless point-to-point interconnections were macromodeled using dimensional analysis. The approach yielded a very simple linear model for lossless lines, and a quadratic model for lossy lines. The lossy line macromodel was used to study the sensitivity of line delay to various circuit parameters.

Finally, we have shown that failure to account for signal transition times in path delay computations can cause significant errors. We proposed two solutions; the first is an extension to the standard fixed-delay CPM algorithm. The second consists of a context-based delay modeling step followed by standard fixed-delay CPM techniques. Both approaches have been shown to predict path delays with a high degree of accuracy.

The static timing analysis in the design of our microprocessors has employed this macromodeling work, and we have made it available to commercial CAD companies.

2.2 GaAs SRAM

A number of small innovations in GaAs RAM design were made in this project, including a method for electronically selecting redundant rows or columns to improve yield. In this scheme, a selection inverter can be programmed by applying a controlled over-voltage waveform at the gate of the inverting MESFET. The device retains a low-impedance from drain to source, producing a permanent logic 0 at the output of the inverter.

A more fundamental development (patent pending) was a new type of memory cell, called the current mirror memory cell. This cell enables the memories to achieve higher speeds

than was previously possible in GaAs. The new cell is also smaller and more reliable than the conventional cell, enabling higher integration levels and better yields than before. In large memories, a 25% improvement in power-delay performance has been seen with this new circuit.

2.3 Circuit Optimization and SRAM Compiler

To make this new memory style easily accessible to other circuit designers, a compiler was developed which automatically generates register files and embedded GaAs memories, based on user specifications, up to 8Kb in size. The compiler includes some unique features that are necessary to achieve good yields in GaAs RAMS, and others which take the compiler beyond the state of the art for conventional CMOS SRAM compilers. It includes the following properties:

1. Performance Driven Transistor Sizing: An algorithm has been devised to method-ically explore the design space of transistor sizing to arrive at a design that in-telligently trades speed for power while attempting to meet a target specification (access and write time). HSPICE (rather than macromodels) is used for delay and power calculations.

2. Process Tolerance: Rather than simulating a design and checking for functionality at the corners after the design is performed, the tool has process tolerance built into the heart of the transistor sizing routine to guide size selection based upon the process spread.

3. Integrated CAD System: The CAD framework that this compiler is built upon is a highly integrated system which re-extracts parasitic Rs and Cs while guiding the memory sizing.

4. Process Independence: Rather than tying the compiler to a processing technology using delay macromodels (as is done with most other RAM compilers), the compiler has been made as general as possible. Inputs to the compiler are: SPICE transistor models files, a process spread file (sigma V_{TE}, V_{TD}, etc.), a sheet resistance file, an inter-layer capacitance file, and a design rule file (for design rule-independent layout).

The result is a framework which be be easily ported to a new process, but more importantly, provides the ability to predict the impact of technology variations on speed, power, and area. The compiler is being licensed for commercial use, and designs done by it are being used to drive integration density on Motorola's GaAs fabrication line.

2.4   Architectural Support for Multiple-API Systems

Software development costs form a significant portion of the total cost of computing systems. Even after initial development, software applications often require additional programming effort to be ported to different operating system platforms. This is required because most existing systems support only a single application programming interface (API), such as that of the Apple Macintosh, Microsoft Windows, or a UNIX variant. Many new operating systems, such as Windows NT, Mach 3.0, Chorus, V and Sprite are designed to minimize these porting costs by supporting multiple APIs. Unfortunately, the additional features of these systems comes with a cost; they are typically slower than traditional systems with a single API.

To mitigate these performance penalties, we have investigated ways of tuning hardware architectures to better support multiple-API systems. This work involves the development of new analysis tools and methodologies that consider operating system effects, as well as studies that compare architectural design trade-offs when supporting operating system

code.

Initial results from this work have shown that multiple-API systems, such as Mach 3.0, tend to stress existing hardware architectures more than single-API systems such as Ultrix. For example, Mach 3.0 exhibits far greater numbers of translation look-aside buffer (TLB) and instruction cache (I-cache) misses. However, through minor, low-cost adjustments to the TLB and I-cache, existing architectures can be re-designed to recover much higher levels of performance.

2.5   Trap-Driven Simulation

The method we have developed and implemented for measuring memory system (TLB and cache) performance is called trap-driven simulation. Our trap-driven simulator, Tapeworm II, is a software-based simulation tool that evaluates the cache and TLB performance of multiple-task- and operating system-intensive-workloads. Tapeworm resides in an OS kernel and causes a host machine's hardware to drive simulations with kernel traps instead of with address traces, as is conventionally done. This allows Tapeworm to quickly and accurately capture complete memory referencing behavior with a limited degradation in overall system performance.

Where miss ratios are reasonable, Tapeworm simulations are significantly faster than traditional trace-driven simulations. Tapeworm typically slows a system down by less than an order of magnitude (10x) when cache miss ratios are under 10%, and slow-downs approach zero as miss ratios decrease. Tapeworm can employ set sampling techniques to further reduce slow-downs, but at the expense of higher measurement variance. Unlike trace-driven simulations, which typically produce identical results from run to run, trap-driven simulations exhibit greater sensitivity to inherent variations in memory system behavior on a real machine. Less than 5% of Tapeworm's code is machine-dependent, enhancing its portability to different machines. Only a few essential primitive operations are required

of a machine to support Tapeworm. Although the trap-driven approach is flexible enough to simulate most TLB and cache configurations, other architectural structures, such as write buffers and instruction pipelines cannot be simulated with this approach. Tapeworm implementations currently exist for TLB and instruction cache simulation on MIPS-based DECstations and for TLB simulation on a 486-based Gateway PC.

2.6 Technology-Organization Trade-off Methodology

A thorough empirical understanding of the interactions between implementation technologies and computer organization is needed to design computers which reach the performance potential of their implementation technologies. We have developed a design methodology that can be used to quantitatively analyze technology-organization interactions and their impact on computer performance. The problem can be viewed as a set of trade-offs between cycle time and cycles per instruction. The product of cycle time and cycles per instruction is time per instruction, which is used as the performance metric in our methodology. Two new analysis tools, a timing analyzer and a trace-driven cache simulator were developed as integral parts of the methodology.

This methodology was used to optimize the design of a two-level cache memory system for a GaAs microprocessor. The results were both a validation of the methodology and a number of useful architectural design guidelines. For example, when the first-level cache has a pipeline depth of one, maximum performance is reached with small caches that have short access times. When the first-level cache is more deeply pipelined, higher performance is reached with larger caches, which have longer access times. The increase in cycles per instruction caused by deeper pipelining can be hidden by using static compiler scheduling techniques.

Using this methodology, we have also made general observations about second-level caches. Second-level caches that are split between instructions and data have perfor-

mance and implementation benefits. The choice of write-policy (write-back or write-through) depends on the effective access time of the next higher level of the memory hierarchy. Adding concurrency to the cache-hierarchy in the form of write-buffering of the second-level cache or a non-blocking first-level cache provides modest reductions in CPI. When the reduction in CPI was weighed against the increased implementation complexity and possible increase in cycle time, these techniques for improving performance were not as effective as increasing the size and pipelining depth of the first-level cache.

2.7   Integrated Circuit Designs

In addition to many test chips, the project has included the design of three gallium arsenide microprocessor chips, named Aurora I, II, and III. The first two of these have been fabricated and tested, showing operation of 100 and 170 MHz, respectively. These chips, which execute subsets of the MIPS instruction set, were fabricated by Vitesse Semiconductor through the ARPA-supported MOSIS service, and tested in the high-speed digital testing facility at the University.

2.7.1   Aurora I CPU

The Aurora I processor was implemented in the Vitesse HGaAs II process. This chip consists of 60,500 transistors, measures 12.175 x 7.941 mm, and dissipates 11 watts. The primary purpose in designing this chip was to exercise the computer-aided design environment that has been developed jointly by the University of Michigan and Cascade Design Automation (Bellevue, WA) for the rapid design of very high-speed processors. The tools in this environment allow specification of the chip using a hardware description language. From this description, simulation models and the physical layout of gallium arsenide circuits are synthesized. Aurora served as a qualification vehicle for both the circuit design and the design methodology. The fact that Aurora was designed by five

graduate students in just five months underscores the success of the design environment.

A manual design error which disabled an off-chip address bus, and a package bonding error combined to limit the testability of this chip. Fortunately, the chip was fully scanned, so that it was possible to verify functionality and timing of almost all of the circuit. The fastest chip (of a small number fabricated) operated at 137 MHz on all of the observable paths. Since the longest path could be to the disabled address bus, the chip's speed must be De-rated.

2.7.2 Aurora II CPU

The Aurora II chip was composed of 160,000 transistors in the Vitesse HGaAs III process. It executes 40 instructions and supports cache memory. The Aurora II illustrates an important advantage of a rapid design environment: it allows designers to quickly capitalize on advances in technology. This chip was designed in a new semiconductor process by 4 students in 6 months. The Aurora II traps on unimplemented instructions to give it MIPS binary compatibility; a C compiler which targets the Aurora II instruction set was also developed.

Two manual design errors in this chip (clock phase errors introduced during hand timing optimization and a short between a power rails over I/O pads) required revision. We also discovered a data translation problem in the path from our tools to masks through MOSIS. These problems were corrected and the design was fabricated again. The resulting circuits were testable and operation at 200 MHz was verified. Yield was low, possibly in part because we were using a superbuffer that was not tolerant of process variations. The Aurora II processor also had an error in the instruction decoding logic, which demonstrated the need for more thorough verification of designs and pointed out the need for certain additional CAD capabilities.

2.7.3 Bus Interface Unit and I/O Pads

We have designed a bus interface unit to transfer 32-b data over a bidirectional bus on both edges of a 330 MHz clock. The chip is composed of 80K transistors, it is 5x8 mm in size, and it dissipates 8 watts. This circuit was designed as part of the Aurora III chipset, to be used as a high-speed interface between the CPU and MMU. It was fabricated as a separate test chip, having 184 pins, 144 of which were signal pins. The first version of this chip had one design error which rendered half of the circuit non-functional. This was corrected and many critical paths were reduced in length on a second version of the chip which has been fabricated and is ready to be tested.

The bus interface unit test chip included programmable I/O pads designed to operate at up to 500 MHz. The pad supports Gunning Transceiver Logic, Emitter Coupled Logic, and Rambus voltage levels. The switching levels can be set manually or automatically, using on-chip digital calibration.

2.7.4 Aurora III CPU

Architectural decisions in the Aurora III chips were based on results of a simulator which was written in our group for the CPU and FPU. This trace-driven simulator runs application code from the SPEC92 benchmark suite. The simulator includes support for: dual issue of integer and floating point instructions; prefetching of integer and floating point data; write buffer to decouple data stores to the memory hierarchy from other instructions; queues to decouple floating-point and integer instructions; instruction issue policies; and variation in sizes of various chip resources, such as queues and reorder buffers. The programs were compiled using GCC with no additional code rescheduling.

The chip designs in this processor are the culmination of four years of research on the effective utilization of gallium arsenide technology for the implementation of RISC mi-

croprocessors. The integer processor consists of five functional modules that operate semi-autonomously to fetch, decode, execute and retire instructions. It includes the bus interface unit described above, an integer execution unit, an instruction fetch unit, a load/store unit, and a prefetch unit for data and instructions. Integer computations require four cycles, while memory instructions require six or seven.

Three machine models, small, baseline and large, were evaluated to study the return on resources required to implement various architectural features. Each model was evaluated with different memory latency assumptions. Based on this study, a superscalar, decoupled, CPU with a non-blocking memory system, out-of-order completion, branch prediction, and speculative execution, was designed in GaAs. The layout includes over 500,000 transistors; it has not been fabricated.

2.7.5 Aurora III Floating Point Accelerator

Issues confronting the designer of floating-point units for high-performance microprocessors were also studied, with emphasis on minimizing floating-point stalls of the integer processor. A synchronization problem exists between the integer and floating-point units that causes the FPU to stall the IPU. This can be overcome through the use of decoupling data and instruction queues, a reorder buffer, and result busses. Increasing the number of queue or reorder buffer entries results in improved performance that cannot be equaled either through pipelining the FPU functional units, or by attempts to reduce floating-point functional unit latency, both of which require a significant increase in resources.

One important class of stall conditions can be addressed by: analyzing memory system characteristics; code scheduling to improve FPU performance on commonly encountered instruction sequences; selection of the FPU instruction and data transfer point in the integer pipeline; and the degree of instruction issue. Instruction issue policies attempt to exploit available parallelism that exists in the instruction stream. Different policies

offer design points which, while achieving similar performance, vary with respect to design complexity and resource requirements. The most promising designs either emphasize the extraction of instruction-level parallelism through greater complexity, or focus on simplicity to increase clock frequency.

Several utilities were created to support the implementation of the high-speed VLSI chips used in the project, and suggestions for an automated approach to performing timing analysis and logic optimization are presented.

The culmination of this work was the design of an IEEE-754 compliant double precision floating-point unit; the chip has 500,000 transistors, and was designed in a 1.0mm GaAs direct-coupled FET logic process. Most of the conclusions regarding architectural optimizations are independent of technology, though a number of tradeoffs in the design were made within the constraints of integration levels, fan-in, fan-out, logic topologies, speed, and power of the GaAs direct-coupled FET logic. The final FPU achieves a high level of performance that exceeds many current leading commercial processors.

2.8 High Speed Circuit Design

A number of CAD capabilities were developed to support the design of high-performance integrated circuits. Among the most significant of these are:

- Modification of a delay calculation utility from Cascade to support a macromodel approach for deriving GaAs gate delays. This macromodeling approach is very efficient, and gives results that are within 10% of Spice.

- The two-phase clocking scheme used for the Aurora designs is subject to type of design error in which a given logic block has inputs derived from both clock phases. A utility was written to report any such instances.

- Several utilities were developed to support the analysis of clock skew. These tools will extract the clock distribution network from a design and create a ready-to-run HSPICE file. The resulting simulations provide information about latch-latch skew and can be represented graphically in a 3d plot of clock transit time versus chip location.

- A levelization utility has been written to support analyze of gate-depth along critical paths. A latch-based design will also allow cycle-stealing from the phase which precedes a critical path. This utility generates 3d histograms of current and previous path depth.

- A post-processing optimization utility, based on a program from Cascade, was developed to recognize common logic patterns that can optimized, propagate constants, and merge logic. The utility was also extended to support buffer selection along critical paths.

2.9 Technology Transition

Several CAD capabilities developed in this project have been commercialized by Cascade Design Automation, a Bellevue, Washington company which has collaborated in this work. Cascade's GaAs circuit compiler resulted from their involvement in this project. They have also commercialized the datapath column placement tool, and are in the process of licensing the SRAM compiler. (Contact Ken Rousseau at ROUSSEAU@ole.cdac.com or Richard Oettel at oettel@ole.cdac.com or either one at 206-643-0200.)

Studies of GaAs logic design styles and process characteristics done in this project have influenced process development at Vitesse and Motorola. Vitesse plans to use designs from this project to evaluate the new HGaAs IV process. Motorola is using SRAMs designed in this project to drive process development. (Contact Ray Milano at Vitesse,

milano@vitsemi.com or 805-388-7541, or Peter Zdebel at Motorola, 602-897-4469.) Cray Computer switched logic families and moved to higher integration levels for the Cray 4 machine as a result of studies based on our collaboration.

Tapeworm, a program and methodology developed in this project for monitoring processor performance while executing operating system routines, is being built into Alpha-based systems at DEC and into IBM systems. (Contact Joel Emer emer@vssad.enet.dec.com.)

3 Publications

1. Jeffrey Alan Dykstra, "High-Speed Microprocessor Design with Gallium Arsenide Very Large Scale Integrated Digital Circuits," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, University of Michigan, 1990.

2. O. Olukotun, R. Brown, R. Lomax, T. Mudge, K. Sakallah, "Multilevel Optimization in the Design of a High-Performance GaAs Microcomputer," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 5, May 1990, pp. 763–767.

3. K. Sakallah, T. Mudge, O. Olukotun, "$checkT_c$ and $minT_c$: Timing Verification and Optimal Clocking of Synchronous Digital Circuits," *ICCAD'90 Digest of Technical Papers*, November 12–15, 1990, pp. 552–555.

4. O. Olukotun, "Technology-Organization Trade-offs in the Architecture of a High Performance Processor," Ph. D. dissertation, University of Michigan, 1991.

5. T. Mudge, R. Brown, W. Birmingham, J. Dykstra, A. Kayssi, R. Lomax, O. Olukotun, K. Sakallah, R. Milano, "The Design of a Micro-Supercomputer," *Computer Magazine*, January 1991, vol. 24, no. 1, pp. 57–64.

6. T. Mudge, R. Brown, W. Birmingham, J. Dykstra, A. Kayssi, R. Lomax, O. Olukotun, K. Sakallah, R. Milano, "The Design of a Micro-Supercomputer," *Proc. of the 24th Annual Hawaii International Conference on System Sciences*, January 8–11, 1991, vol. 1, pp. 421–432.

7. A. I. Kayssi, K. A. Sakallah, R. B. Brown, R. J. Lomax, T. N. Mudge, T. R. Huff, "Impact of MCM's on System Performance," *1991 Multichip Module Workshop: Extended Abstract Volume*, Santa Cruz, CA, March 28 – March 29, 1991, pp. 58–65.

8. R. B. Brown, "The Silicon Compiler as a Tool for Teaching Microelectronic System Design," *Technical Digest of the Third Physical Design Workshop*, Laurel Highlands, PA, May 20–22, 1991, pp. 64–75.

9. M. Upton, T. Hoy, T. Huff, R. Brown, R. Oettel, M. Rossman, "Module Generation for a GaAs Microprocessor," *Technical Digest of the Third Physical Design Workshop*, Laurel Highlands, PA, May 20–22, 1991, pp. 88–94.

10. O. A. Olukotun, T. N. Mudge, R. B. Brown, "Implementing a Cache for a High-Performance GaAs Microprocessor," *The 18th Annual International Symposium on Computer Architecture*, vol. 19, no. 3, May, 1991, pp. 138–147.

11. R. Brown, A. Chandna, T. Hoy, T. Huff, D. Johnson, R. Lomax, T. Mudge, D. Nagle, O. Olukotun, K. Sakallah, R. Uhlig, M. Upton, "Synthesis and Verification of a GaAs Microprocessor from a Verilog Hardware Description," *Proceedings Open Verilog International User Group Meeting*, March 1992, pp. 85–93.

12. K. Sakallah, T. Mudge, O. Olukotun, "Analysis and Design of Latch-Controlled Synchronous Digital Circuits," *IEEE Transactions on CAD of IC's and Systems*, vol. 11, no. 3, March 1992, pp. 332–333.

13. A. Kayssi, K. Sakallah, R. Brown, R. Lomax, T. Mudge, T. Huff, "Impact of MCM's on System Performance Optimization," *1992 IEEE International Symposium on Circuits and Systems*, San Diego, CA, May 1992, vol. 2, pp. 919–922.

14. D. Nagle, R. Uhlig, T. Mudge, "Monster: A Tool for Analyzing the Interaction Between Operating Systems and Computer Architecture," Technical Report CSE-TR-147-92, University of Michigan, Department of Electrical Engineering and Computer Science, May 1992.

15. K. Olukotun, T. Mudge, R. Brown, "Performance Optimization of Pipelined Primary Caches," *The 19th Annual International Symposium on Computer Architecture*, Gold Coast, Australia, May 19–21, 1992, vol. 20, no. 2, pp. 181–190.

16. A. I. Kayssi, K. A. Sakallah, "Delay Macromodels for the Timing Analysis of GaAs DCFL," *Proc. European Design Automation Conference (EuroDAC)*, Hamburg, Germany, September 1992, pp. 142–145.

17. R. Brown, A. Chandna, T. Huff, R. Lomax, T. Mudge, R. Oettel, M. Upton, "Compound Semiconductor Device Requirements for VLSI," *Proceedings of the 19th International Symposium on Gallium Arsenide and Related Compounds 1992: Institute of Physics Conference Series Number 129*, Karuizawa, Japan, October 1992, pp. 857–862.

18. R. Brown, P. Barker, A. Chandna, T. Huff, R. Lomax, T. Mudge, K. Sakallah, P. Sherhart, R. Uhlig, M. Upton, "GaAs RISC Processors," *GaAs IC Symposium*, Miami, Oct. 1992, pp. 81–84.

19. M. Golden, T. Mudge, "Hardware Support for Hiding Cache Latency," Technical Report CSE-TR-152-93, University of Michigan, Department of Electrical Engineering, Jan. 1993.

20. M. Upton, T. Huff, P. Sherhart, P. Barker, R. McVay, T. Stanley, R. Brown, R. Lomax, T. Mudge, and K. Sakallah, "A 160,000 Transistor GaAs Microprocessor," in *International Solid-State Circuits Conference Digest of Technical Papers*, vol. 36, Feb. 1993, pp. 92–93.

21. A. Chandna and R. Brown, "A 32kb GaAs SRAM with Electronically Programmable Redundancy," *Research on Integrated Systems: Proceedings of the 1993 Symposium*, Seattle, WA, March, 1993, pp. 155–167.

22. T. Huff, M. Upton, P. Sherhart, P. Barker, R. McVay, T. Stanley, R. Brown, R. Lomax, T. Mudge, and K. Sakallah, "A High Performance GaAs Microprocessor," *Proceedings of the IEEE Laser and Optics Society Sarnoff Symposium*, Princeton, N. J., March 1993.

23. A. I. Kayssi, K. A. Sakallah, T. N. Mudge, "The Impact of Signal Transition Time on Path Delay Computation," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 40, no. 5, May 1993, pp. 302–309.

24. D. Nagle, R. Uhlig, T. Stanley, S. Sechrest, T. Mudge, and R. Brown, "Design Tradeoffs for Software-Managed TLBs," *Proceedings of the 20th Annual International Symposium on Computer Architecture*, May 1993, pp. 27–38.

25. T. Huff, M. Upton, T. Mudge, R. Brown, "The Aurora Project," *Record of Hot Chips V*, August 1993, pp. 3.2.1 - 3.2.12.

26. K. Sakallah, T. N. Mudge, T. Burks, E. S. Davidson, "Synchronization of Pipelines," *IEEE Trans. on CAD Integrated Circuits and Systems*, vol. 12, no. 8, August 1993, pp. 1132–1146.

27. A. Chandna, R. B. Brown, "An Asynchronous GaAs MESFET Static RAM Using A New Current Mirror Memory Cell," *15th Annual GaAs Symposium: Technical Digest 1993*, Oct. 1993, pp. 79–82.

28. A. Kayssi, K. Sakallah, "Delay Modeling for GaAs DCFL Circuits," *15th Annual GaAs Symposium: Technical Digest 1993*, Oct. 1993, pp. 67–70.

29. R. Brown, M. Upton, A. Chandna, T. Huff, T. Mudge, R. Oettel, "Gallium Arsenide Process Evaluation Based on a RISC Microprocessor Example," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 10, Oct. 1993, pp. 1030–1037.

30. J. C. Toole, R. B. Brown, "GaAs Opportunities in High Performance Computing and Communications," *15th Annual GaAs Symposium: Technical Digest 1993*, Oct. 1993, pp. 3–6.

31. T. Stanley, M. Upton, P. Sherhart, T. Mudge, R. Brown, "A Microarchitectural Performance Evaluation of a 3.2 Gbyte/s Microprocessor Bus," *MICRO-26: Proceedings of the 26th Annual International Symposium on Microarchitecture*, Dec. 1993, pp. 31–40.

32. D. Nagle, R. Uhlig, T. Mudge, S. Sechrest, "Optimal Allocation of On-chip Memory for Multiple-API Operating Systems," *Proceedings of the 21st Annual International Symposium on Computer Architecture*, April 18–21, 1994, pp. 358–369.

33. A. Kayssi, K. Sakallah, "Delay Macromodels for Point-to-Point MCM Interconnections," *IEEE Transactions on Components, Packaging and Manufacturing Technology - Part B: Advanced Packaging*, vol. 17, no. 2, May 1994, pp. 147–152.

34. R. Uhlig, D. Nagle, T. Mudge, S. Sechrest, "Kernel-based Memory Simulation," *Proceedings: ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, vol. 22, no. 1, May 1994, pp. 286–287.

35. A. Kayssi, K. Sakallah, "Macromodel Simplification Using Dimensional Analysis," *1994 IEEE International Symposium on Circuits and Systems*, vol. 1, May 30-June 2, 1994, pp. 335–338.

36. R. Uhlig, D. Nagle, T. Mudge, S. Sechrest, "Trap-driven Simulation with Tapeworm II," *ASPLOS VI Proceedings: Sixth International Conference on Architectural Support for Programming Languages and Operating Systems*, San Jose, CA, Oct. 4–7, 1994, pp. 132–144.

37. M. Upton, T. Huff, T. Mudge, R. Brown, "Resource Allocation in a High Clock Rate Microprocessor," *ASPLOS VI Proceedings: Sixth International Conference on Architectural Support for Programming Languages and Operating Systems*, San Jose, CA, Oct. 4–7, 1994, pp. 98–109.

38. P. Sherhart, M. Upton, R. Lomax, R. Brown, "A Variable-Voltage Bidirectional I/O Pad for Digital GaAs Applications," *16th Annual IEEE GaAs IC Symposium: Technical Digest 1994*, Oct. 16–19, 1994, pp. 67–70.

39. A. Chandna, R. B. Brown, D. Putti, C. D. Kibler, "Power Rail Logic: A Low Power Logic Style for Digital GaAs Circuits," *16th Annual IEEE GaAs IC Symposium: Technical Digest 1994*, Oct. 16–19, 1994, pp. 71–74.

40. A. Chandna, R. B. Brown, "An Asynchronous GaAs MESFET Static RAM Using A New Current Mirror Memory Cell," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 10, Oct. 1994, pp. 1270–1276.

41. M. Golden, T. Mudge, "A Comparison of Two Pipeline Organizations," *Micro-27: Proceedings of the 27th Annual International Symposium on Microarchitecture*, San Jose, CA, Nov. 30–Dec. 2, 1994, pp. 153–161.

42. Daniel Kaiser, "Loop Optimization Techniques on Multi-Issue Architectures," Ph. D. dissertation, Department of Electrical Engineering and Computer Science, University of Michigan, 1995.

43. Richard A. Uhlig, "Trap-driven Memory Simulation," Ph. D. dissertation, Department of Electrical Engineering and Computer Science, University of Michigan, 1995.

44. Michael D. Upton, "Architectural Trade-offs in a Latency Tolerant Gallium Arsenide Microprocessor," Ph. D. dissertation, Department of Electrical Engineering and Computer Science, University of Michigan, 1995.

45. Ayman I.. Kayssi, Karem A. Sakallah, "Timing Models for Gallium Arsenide Direct-Coupled FET Logic Circuits," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 3, March 1995, pp. 384–393.

Articles Accepted for Publication

- B. Davis and T. Mudge, "A Verilog Preprocessor for Representing Datapath Components," *1995 International Verilog HDL Conference*, to appear.

- D. Nagle, R. Uhlig, T. Stanley, T. Mudge, S. Sechrest, R. Brown, "Design Tradeoffs for Software-Managed TLBs," *ACM Transactions on Computer Systems*, to appear.

- K. Sakallah, T. Burks, T. Mudge, "Critical Paths in Circuits with Level-Sensitive Latches," *IEEE Transactions on VLSI Systems*, to appear.

- K. Sakallah, M. A. Riepe, J. P. Silva, R. B. Brown, "Ravel-XL: A Hardware Accelerator for Assigned-Delay Compiled-Code Logic Gate Simulation," *IEEE Transactions on VLSI Systems*, to appear.

- T. J. Stanley, T. N. Mudge, "A Systematic Approach to Multi-objective Computer Architecture Optimization," *1995 Conference on Advanced Research in VLSI*, to appear.

4 Personnel

Faculty

Richard B. Brown
Trevor N. Mudge
Ronald J. Lomax
Karem A. Sakallah
William P. Birmingham

Students	Degree	Professional Affiliation
Phillip J. Barker	M. S.	Advanced Micro Devices, Austin, TX
Ajay Chandna	Ph. D.	Cascade Design Automation, Phoenix, AZ
Chin-Po Chen	M. S.	
Michael L. Golden		
Thomas A. Hoy	M. S.	Motorola MMTG, Austin, TX
Thomas R. Huff	Ph. D.	Intel, Hillsboro, OR
Bruce L. Jacob		
David M. Johnson	M. S.	Cascade Design Automation, Bellevue, WA
Daniel Kaiser	Ph. D.	Cadence, Boston, MA
Ayman I. Kayssi	Ph. D.	Faculty, American University of Beirut, Lebanon
C. David Kibler	M. S.	Hewlett-Packard, Ft. Collins, CO
Robert G. McVay	M. S.	Lockhead-Sanders, Nashua, NH
Mark Michmerhuizen	M. S.	Smith Industries, Grand Rapids, MI
David F. Nagle	Ph. D.	Faculty, Carnegie Mellon University, Pittsburg, PA
Kunle A. Olukotun	Ph. D.	Faculty, Stanford University, Stanford, CA
David D. Putti	M. S.	Motorola MMTG, Austin, TX
Michael A. Riepe		
Mark Robert	M. S.	NexGen, Santa Clara, CA
Shailendra A. Save	M. S.	Cascade Design Automation, Bellevue, WA
Patrick J. Sherhart	M. S.	Motorola MMTG, Austin, TX
Timothy J. Stanley		
Richard A. Uhlig	Ph. D.	Post-doctoral Fellowship, Europe
Michael D. Upton	Ph. D.	Intel, Hillsboro, OR

5 Report of Inventions

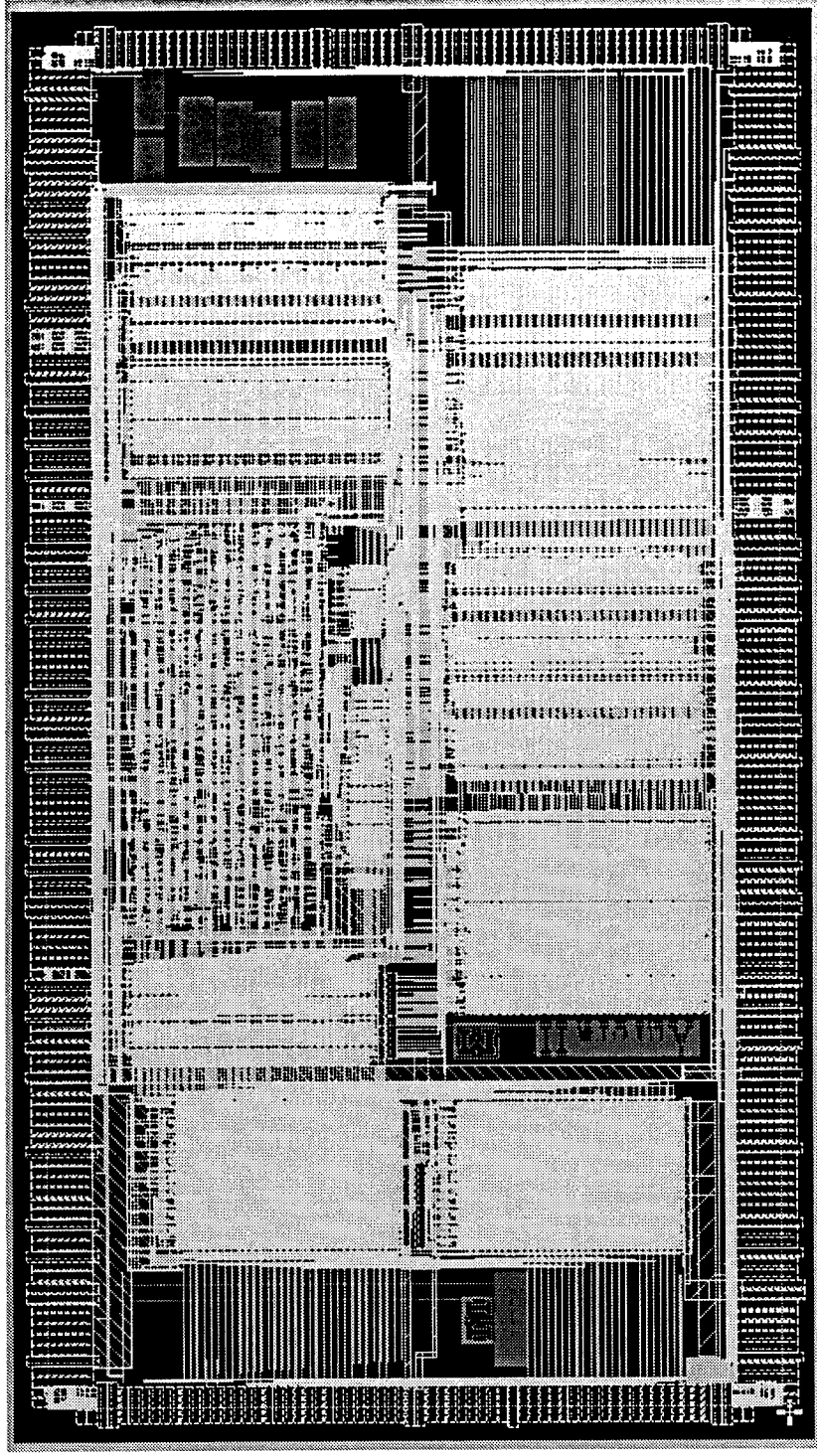High Speed Current Mirror Memory Cell Architecture. (UM File #925.)

# 6 Appendix: Chip Layouts

Aurora I CPU
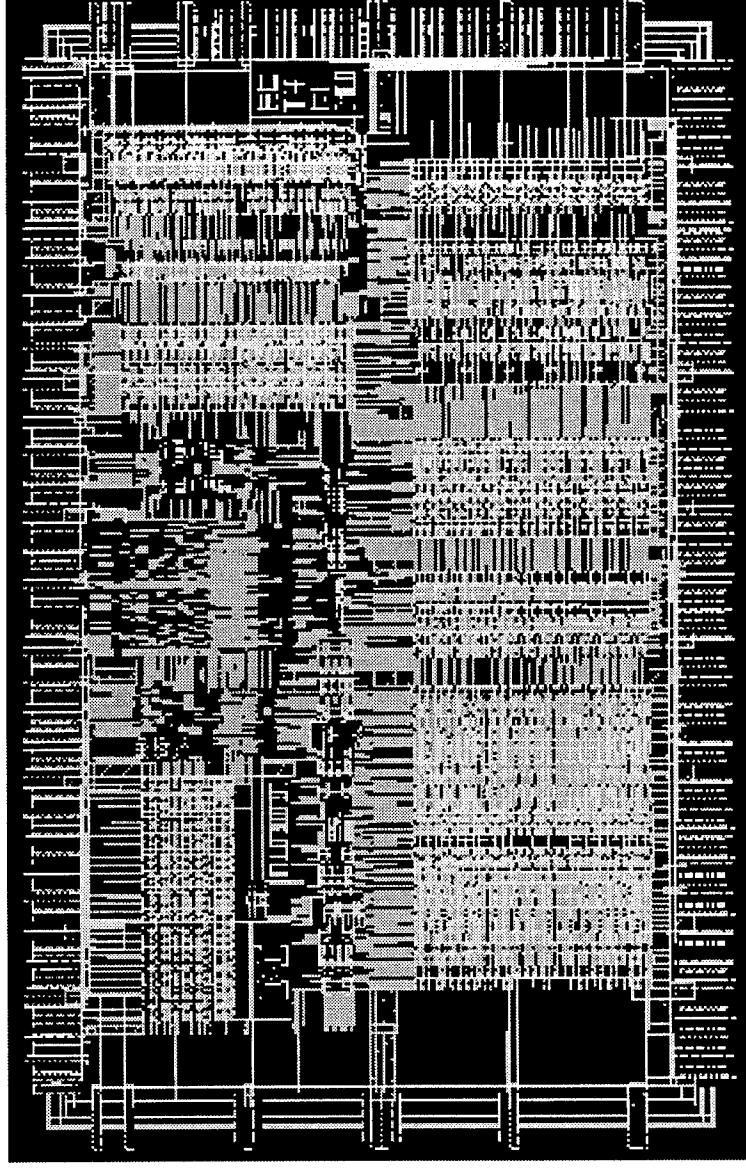
- MIPS Architecture
- 100 MHz
- 65,000 Transistors
- 30 Instructions
- Vitesse HGaAs II
- Modified Ling Adder
- 3-port 32x32 RF
- 11 Watts
- 20% Yield
- Fully Scanned
- One Known Bug
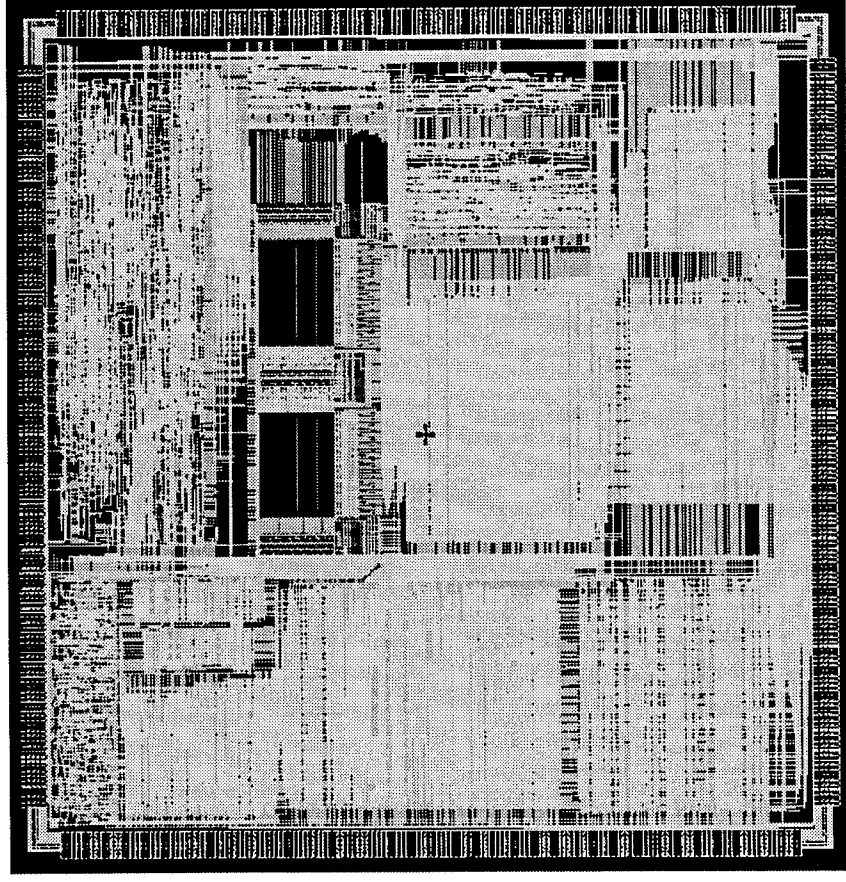- 5 Students / 5 mo.

Aurora II CPU

- 160 K Transistors
- 180 MHz, 20 Watts

- Vitesse HGaAs III
- 40 MIPS Instructions
- Traps on Unimplemented Instructions
- Primary and Secondary Cache Support

- 1.6 ns 32 x 32 3-port RF
- 1.7 ns 32-bit Adder
- 5 students x 1 yr
- Two functional errors

Bus Interface Unit
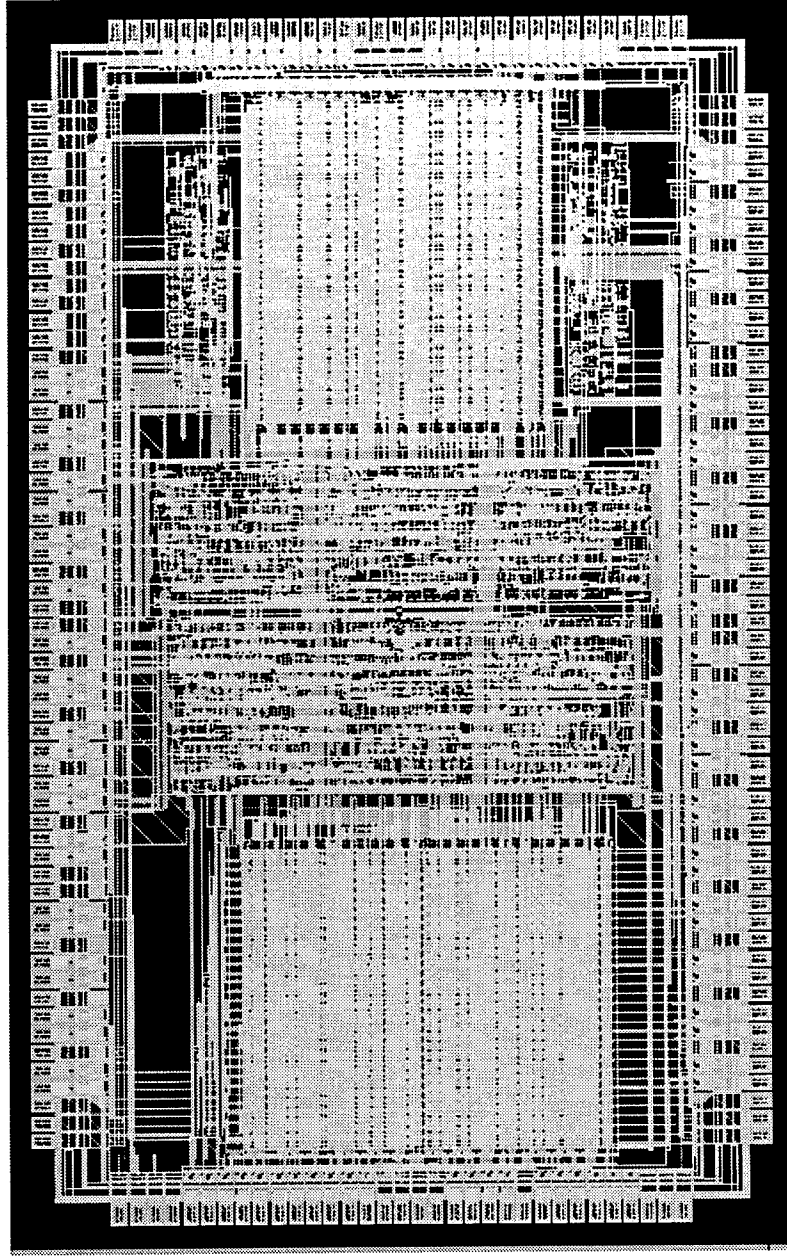
- 330 MHz
- 80 K Transistors
- 8 Watts

- 32-bit Bidirectional Bus
- Synchronous Dual-Edge Transfers
- 1 Student / 1 yr.

Aurora III Floating-Point Unit

- IEEE-754 Compliant
- 40 Instructions
- 500 K Transistors
- 16 x 16 mm
- 250 MHz, 300 SPECfp92
- Precise and Real-Time Modes
- Decoupled Architecture
- 5-Cycle Multiply
- 3-Cycle Add
- 2-Cycle Conversion Unit
- Data Prefetching
- 5 Students x 1.5 yr.